EXPRESS MAIL LABEL NO.: EV019279539US DATE OF DEPOSIT: DECEMBER 11, 2003

I hereby certify that this paper and fee are being deposited with the United States Postal Service Express Mail Post Office to Addressee service under 37 CFR § 1.10 on the date indicated below and is addressed to the Mail Stop Patent Application, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450

VENESSA M. URENA
NAME OF PERSON MAILING PAPER AND FEE _ SIGNATURE OF PERSON MAILING PAPER AND FEE

Inventor(s): Jeffrey Chase
Ronald P. Doyle

AUTONOMIC SELECTION OF A REQUEST ROUTING POLICY BASED UPON CACHE EFFECTIVENESS

BACKGROUND OF THE INVENTION

Statement of the Technical Field

[0001] The present invention relates to the request routing in a content delivery network and more particularly to the autonomic selection of a routing policy based upon the predicted cache effectiveness of the selected routing policy.

Description of the Related Art

[0002] In the prototypical content delivery system, content can be delivered from an origin server to a community of content consuming clients. Content typically can be delivered according to a request-response paradigm in which the content consuming clients initiate a request for content to which one or more origin servers can respond with the requested content. Generally, one or more content caches can be disposed in the intermediate communications path between the content consuming clients and content servers in order to enhance the responsiveness of the servers to any single client request and to reduce the processing burden placed upon the origin server.

[0003] A variety of mechanisms route content request streams through intermediate caches. For instance, content clients may be configured to use a particular cache. Similarly, the content delivery system itself may redirect requests by interposing on DNS translations or by intercepting requests at the IP level. In addition, each cache may control the routing of its own miss stream to other components. The last two years have seen an explosion of growth in content caching and content delivery infrastructure. Key developments include the increased role of surrogate caching among hosting providers and the aggregation of content consumers into large Internet service providers employing transparent interception proxies based upon Layer 7 switches. These developments have fed the growth in demand for Web caching systems.

[0004] Today, server farms host many content sites, where a group of servers can be clustered together to act as a unified server to external clients. Any given request could be handled by any of several servers, thereby improving scalability and fault-tolerance. The switching infrastructure connecting the servers to the hosting network generally includes one or more redirecting server switches to route incoming request traced to the servers. Referred to in the art as request distributors, these switches select individual servers to handle each incoming content request. Thus, the server selection policy can play an important role in managing cluster resources in order to maximize throughput and meet quality-of-service goals.

[0005] Conventional server switches often incorporate a variety of request routing methodologies when distributing requests to backend server processes. In particular, the server selection methodologies can be selected in order to maximize throughput and minimize response latency. For instance, server load balancing oriented

methodologies monitor server status and direct requests to lightly loaded servers. Notably, server load balancing switches often are referred to as Layer 4 switches because server load balancing switches make server selection decisions at connection setup time, and examine only the Layer 4 transport headers of the incoming packet stream.

[0006] Content-aware server selection policies, by comparison, prefer servers that can handle a given request most efficiently. Importantly, the most efficient requesting handling servers incorporate caching technology and, accordingly, the server most likely to be able to process a request most effectively is the server likely to have the requested data in cache. Uniform Resource Locator (URL) hashing is a content-based policy that applies a simple deterministic hash function upon the request URL to select a server. URL hashing has often been referred to as a Layer 7 policy because the URL hashing switch typically parses protocol headers at Layer 7 in order to extract the respective URL.

[0007] Observations of content request patterns drive the design choices and policies for all of these components of a content delivery architecture. In particular, a number of studies indicate that requests to retrieve static Web objects follow a Zipf-like popularity distribution. Specifically, in accordance with Zipf, the probability $\mathbf{p}_i$ of a request for the $\mathbf{i}^{th}$ most popular document is proportional to $\mathbf{1/i}^{\alpha}$ for some parameter α. In this Zipf-like distribution, a large number of object requests typically target the most popular object sources and the most popular objects within those sources. The Zipf-like distribution, however, also includes a long, heavy tail of less popular objects with poor reference locality. Notably, higher α values increase the concentration of requests on the most

popular objects. One implication of the Zipf-like behavior of the Web is that caching is highly effective for the most popular static, and thus cacheable objects, assuming that popularity dominates rate of change. Unfortunately, caching is less effective in respect to the heavy tail of the distribution, which comprises a significant fraction of requests. Hence, Web cache effectiveness typically improves only logarithmically with the size of the cache, measured either by capacity or by user population.

[0008] Zipf-like behavior also has implications for selecting a request routing policy in a server cluster. For example, the Zipf-like behavior of the Web creates a tension between the competing goals of load balancing and locality. On the one hand, content-aware policies such as Layer 7 URL hashing effectively take advantage of the locality present in the request stream by preferring the same server for repeat requests, maximizing server memory hits for popular objects. However, Layer 7 URL hashing remains vulnerable to load imbalances because the most popular objects receive the largest number of requests, and a single server handles all requests for any given object. Layer 4 type server load balancing policies balance load, but Layer 4 type server load balancing policies tend to scatter requests for each object across the servers, reducing server memory hits in the cache for moderately popular objects.

[0009] Recent research has studied this tradeoff in depth, and has resulted in the development of the Locality Aware Request Distribution policy and related policies to balance these competing goals, combining the benefits of each approach. Other commercial request distributors use less sophisticated strategies such as assigning multiple servers to each URL hash bucket, and selecting from the target set using load information. In either case, however, the skilled artisan will recognize the importance of

selecting a suitable routing policy at design time. Accordingly, the conventional selection of a particular routing policy often can depend upon the goals of the systems architect when the system is configured. Predicting the actual requirements of the system at design time, however, can be difficult for most. Moreover, whereas optimally selecting a suitable request routing policy can be problematic generally, in an autonomic system, the problem can be particularly acute.

[0010] For the uninitiated, autonomic computing systems self-regulate, self-repair and respond to changing conditions, without requiring any conscious effort on the part of the computing system operator. To that end, the computing system itself can bear the responsibility of coping with its own complexity. The crux of autonomic computing relates to eight principal characteristics:

I. The system must "know itself" and include those system components which also possess a system identify.

II. The system must be able to configure and reconfigure itself under varying and unpredictable conditions.

III. The system must never settle for the status quo and the system must always look for ways to optimize its workings.

IV. The system must be self-healing and capable of recovering from routine and extraordinary events that might cause some of its parts to malfunction.

V. The system must be an expert in self-protection.

VI. The system must know its environment and the context surrounding its activity, and act accordingly.

VII. The system must adhere to open standards.

VIII. The system must anticipate the optimized resources needed while keeping its complexity hidden from the user.

Thus, in keeping with the principles of autonomic computing, request routing methodologies ought to change as the impact of selecting any one methodology over the other becomes more advantageous for the operation of the system.

SUMMARY OF THE INVENTION

[0011] The present invention addresses the deficiencies of the art in respect to request routing in an information system and provides a novel and non-obvious method, system and apparatus for selecting a request routing policy based upon the cache effectiveness of the selected policy. In a preferred aspect of the invention, an autonomic request routing policy selection system can include a multiplicity of pre-configured request routing policies and a data store of cache metrics for the pre-configured request routing policies. A routing policy selector can be configured for communicative linkage to a server cluster having one or more servers and programmed to select a particular one of the request routing policies for use in routing content requests in the server cluster based upon the cache metrics. In this regard, the routing policy selector further can include a coupling to the routing policies and the data store of cache metrics.

[0012] In a preferred aspect of the invention, the pre-configured request routing policies can include a Layer 4 request routing policy and a Layer 7 request routing policy. The Layer 4 request routing policy can be a server load balancing type policy. By comparison, the Layer 7 request routing policy can be a content localizing type policy. Moreover, the content localizing type policy can be a URL hashing policy. Finally, the cache metrics can include a plurality of Zipf-like analyses based upon different selected alpha values for different workloads imposed upon the server cluster according to different ones of the request routing policies.

[0013] An autonomic request routing policy selection method can include the steps of identifying a contemporary trace footprint experienced by a coupled server cluster, identifying a cache allocation for the coupled server cluster and retrieving at least two sets of hit rate metrics, where each set of metrics corresponds to a particular routing policy. The hit rate metrics can be compared based upon the identified trace footprint and the identified cache allocation to determine a preferred routing policy. As a result, a preferred routing policy can be selected for use in routing content requests to the server cluster. Moreover, an optimal server cluster configuration can be computed with the hit rate metrics for the preferred routing policy and an optimal number of servers can be provisioned in the server cluster based upon the computed optimal server cluster configuration.

[0014] Additional aspects of the invention will be set forth in part in the description which follows, and in part will be obvious from the description, or may be learned by practice of the invention. The aspects of the invention will be realized and attained by means of the elements and combinations particularly pointed out in the appended claims. It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory only and are not restrictive of the invention, as claimed.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] The accompanying drawings, which are incorporated in and constitute part of this specification, illustrate embodiments of the invention and together with the description, serve to explain the principles of the invention. The embodiments illustrated herein are presently preferred, it being understood, however, that the invention is not limited to the precise arrangements and instrumentalities shown, wherein:

[0016] Figure 1 is a schematic illustration of a content distribution server cluster configured with an autonomic routing policy selector which has been configured in accordance with the present invention; and,

[0017] Figure 2 is a flow chart illustrating a process for selecting a routing policy based upon predicted cache effectiveness for the selected routing policy.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0018] The present invention is a method, system and apparatus for autonomically selecting a request routing methodology based upon the cache effectiveness of the selected methodology. In accordance with the present invention, the hit rates for a variable number of servers in a server cluster can be computed for a varying volume of network requests referred to as the trace footprint based upon a selected cache size for the cluster. Notably, the hit rates can be computed for two or more request routing selection policies.

[0019] Subsequently, as the trace footprint experienced within a server cluster changes, the trace footprint and available cache size can be used to select a suitable routing policy. Moreover, once a routing policy has been selected, a suitable number of servers can be provisioned within the server cluster to optimize the cache effectiveness of the system. For example, where the current setting of the size of the cache approaches a significant proportion of the trace footprint, a load balancing oriented request routing policy can be chosen. In contrast, where the trace footprint is large, or where the cache size is small, a content localizing request routing policy can be chosen.

[0020] Figure 1 is a schematic illustration of a content distribution server cluster 110A, 110B, 110n configured with an autonomic routing policy selector 200 which has been configured in accordance with the present invention. The server cluster 110A, 110B, 110n can include server processes and server devices programmed to process requests for content 130, to retrieve requested content from fixed storage and to serve the retrieved content to requesting client processes. Importantly, to vastly enhance the

performance and responsiveness of the server cluster 110A, 110B, 110n, each server process and server device can include a cache in which frequently accessed content can be stored for quick retrieval. As it will be understood by the skilled artisan, the very responsiveness of the server cluster 110A, 110B, 110n can depend upon the effective use of the cache.

[0021] A routing policy selector 200 can be coupled to the server cluster 110A, 110B, 110n and can manage the methodology utilized to route particular content requests to selected ones of the server processes and server devices in the server cluster 110A, 110B, 110n. In this regard, the routing policy selector 200 can include a coupling to a set of routing policies 120A, 120B, 120n. These routing policies can include a range of policies which vary from pure server load balancing oriented policies, to content localizing policies, to an intermediate blend of both. Significantly, the routing policy selector 200 can be programmed to select a particular one of the routing policies 120A, 120B, 120n based upon both computed and observed cache metrics 140.

[0022] The cache metrics 140 can include observed hit rates for specific workloads for each of the routing policies 120A, 120B, 120n. The observed hit rates can reflect a particular cache size expressed as a proportion of available cache memory in the server cluster 110A, 110B, 110n. The hit rates further can vary based upon the number of server processes and devices provisioned for use in responding to the specific workloads in the server cluster 110A, 110B, 110n. Based upon the observed hit rates for the cache metrics 140, the routing policy selector 200 not only can select a suitable one of the routing policies 120A, 120B, 120n for a contemporary trace footprint, but also the routing policy selector 200 can provision a suitable number of server processes and

server devices in the server cluster 110A, 110B, 110n to optimize the cache effectiveness of the server cluster 110A, 110B, 110n based upon the cache metrics 140.

[0023] In more particular illustration, Figure 2 is a flow chart depicting a process for selecting a routing policy based upon predicted cache effectiveness for the selected routing policy. Beginning in block 210, a contemporary trace footprint can be identified as contemporarily experienced in the server cluster. In block 220, a contemporarily configured cache allocation further can be identified. In block 230, previously computed metrics for the server cluster can be retrieved for analysis. The previously computed metrics can include a Zipf-like analysis of previously observed workloads imposed upon varying configurations of the server cluster, but in terms of a number of provisioned servers in the cluster, and also the proportional cache allocation for the cluster.

[0024] Notably, the Zipf-like analysis can be performed for several different alpha values as is well-known in the art and described in substantial depth in R. Doyle, J. Chase, S. Gadde and A. Vahdat, The Trickle-Down Effect: Web Caching and Server Request Distribution, Proceedings of the 6th International Workshop on Web Caching and Content Distribution (WCW '01) (June 2001). The Zipf-like analysis further can be performed for different ones of the request routing policies, including a Layer 4 server load balancing policy and a Layer 7 content localization policy. In block 240, the metrics for the analyses can be compared and in decision block 250 an optimal request routing policy can be selected.

[0025] For instance, where the trace footprint is very large, or where the available cache allocation is quite small, the metrics will indicate a preference for a Layer 7 type routing policy. By comparison, where the cache allocation approaches the same size as the trace footprint, a Layer 4 type routing policy can be preferred. In any case, either a Layer 4 type policy can be selected in block 260, or a layer 7 type policy can be selected in block 270. In either case, once a particular routing policy has been selected, the metrics once again can be consulted to identify an optimal number of servers to be deployed in the server cluster. Noting that the impact of each additional server can fall off logarithmically, it is preferred that a minimum number of servers to achieve optimal cache performance can be provisioned. Once determined, the optimal server configuration can be deployed in block 280.

[0026] The present invention can be realized in hardware, software, or a combination of hardware and software. An implementation of the method and system of the present invention can be realized in a centralized fashion in one computer system, or in a distributed fashion where different elements are spread across several interconnected computer systems. Any kind of computer system, or other apparatus adapted for carrying out the methods described herein, is suited to perform the functions described herein.

[0027] A typical combination of hardware and software could be a general purpose computer system with a computer program that, when being loaded and executed, controls the computer system such that it carries out the methods described herein. The present invention can also be embedded in a computer program product, which comprises all the features enabling the implementation of the methods described

herein, and which, when loaded in a computer system is able to carry out these methods.

[0028] Computer program or application in the present context means any expression, in any language, code or notation, of a set of instructions intended to cause a system having an information processing capability to perform a particular function either directly or after either or both of the following a) conversion to another language, code or notation; b) reproduction in a different material form. Significantly, this invention can be embodied in other specific forms without departing from the spirit or essential attributes thereof, and accordingly, reference should be had to the following claims, rather than to the foregoing specification, as indicating the scope of the invention.